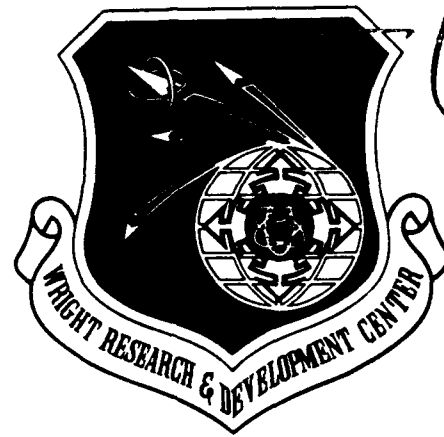


WRDC-TR-90-8007
Volume VIII
Part 11

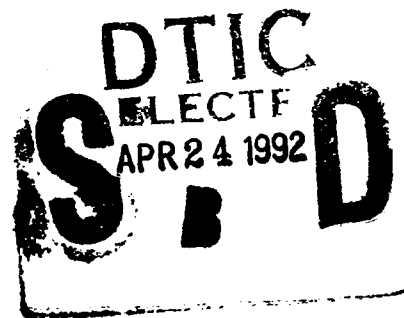
AD-A248 983

INTEGRATED INFORMATION SUPPORT SYSTEM (IISS)
Volume VIII - User Interface Subsystem
Part 11 - Virtual Terminal Development Specification

S. Barker

Control Data Corporation
Integration Technology Services
2970 Presidential Drive
Fairborn, OH 45324-6209



September 1990

Final Report for Period 1 April 1987 - 31 December 1990

Approved for Public Release; Distribution is Unlimited

MANUFACTURING TECHNOLOGY DIRECTORATE
WRIGHT RESEARCH AND DEVELOPMENT CENTER
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433-6533

92-10481



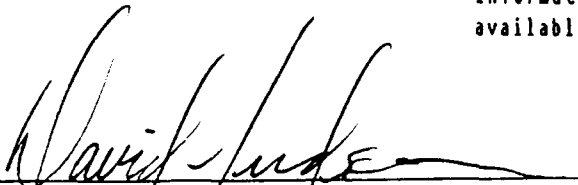
92 4 23 026

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever, regardless whether or not the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data. It should not, therefore, be construed or implied by any person, persons, or organization that the Government is licensing or conveying any rights or permission to manufacture, use, or market any patented invention that may in any way be related thereto.


This technical report has been reviewed and is approved for publication.

This report is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations


DAVID L. JUDSON, Project Manager
WRDC/MTI
Wright-Patterson AFB, OH 45433-6533

25 July 91
DATE

FOR THE COMMANDER:


BRUCE A. RASMUSSEN, Chief
WRDC/MTI
Wright-Patterson AFB, OH 45433-6533

25 July 91
DATE

If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify WRDC/MTI, Wright-Patterson Air Force Base, OH 45433-6533 to help us maintain a current mailing list.

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified		1b. RESTRICTIVE MARKINGS None	
2a. SECURITY CLASSIFICATION AUTHORITY		3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for Public Release; Distribution is Unlimited.	
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE			
4. PERFORMING ORGANIZATION REPORT NUMBER(S) DS 620344300		5. MONITORING ORGANIZATION REPORT NUMBER(S) WRDC-TR-90-8007 Vol. VIII, Part 11	
6a. NAME OF PERFORMING ORGANIZATION Control Data Corporation; Integration Technology Services	6b. OFFICE SYMBOL (if applicable) WRDC/MTI	7a. NAME OF MONITORING ORGANIZATION WRDC/MTI	
6c. ADDRESS (City, State, and ZIP Code) 2970 Presidential Drive Fairborn, OH 45324-6209		7b. ADDRESS (City, State, and ZIP Code) WPAFB, OH 45433-6533	
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Wright Research and Development Center, Air Force Systems Command, USAF	8b. OFFICE SYMBOL (if applicable) WRDC/MTI	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUM. F33600-87-C-0464	
8c. ADDRESS (City, State, and ZIP Code) Wright-Patterson AFB, Ohio 45433-6533		10. SOURCE OF FUNDING NOS.	
11. TITLE Virtual T See block 19		PROGRAM ELEMENT NO. 78011F	PROJECT NO. 595600
		TASK NO. F95600	WORK UNIT NO. 20950607
12. PERSONAL AUTHOR(S) Structural Dynamics Research Corporation: Barker, S.			
13a. TYPE OF REPORT Final Report	13b. TIME COVERED 4 / 1 / 87 - 12 / 31 / 90	14. DATE OF REPORT (Yr., Mo., Day) 1990 September 30	15. PAGE COUNT 40
16. SUPPLEMENTARY NOTATION WRDC/MTI Project Priority 6203			
17. COSATI CODES		18. SUBJECT TERMS (Continue on reverse if necessary and identify block no.)	
FIELD	GROUP	SUB GR.	
1308	0905		
19. ABSTRACT (Continue on reverse if necessary and identify block number)			
<p>This specification establishes the performance, development, test, and qualification requirements of the Virtual Terminal protocol and the computer programs implementing it.</p> <p>BLOCK 11:</p> <p>INTEGRATED INFORMATION SUPPORT SYSTEM Vol VIII - User Interface Subsystem</p> <p>Part 11 - Virtual Terminal Development Specification</p>			
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED x SAME AS RPT. DTIC USERS		21. ABSTRACT SECURITY CLASSIFICATION Unclassified	
22a. NAME OF RESPONSIBLE INDIVIDUAL David L. Judson	22b. TELEPHONE NO. (Include Area Code) (513) 255-7371	22c. OFFICE SYMBOL WRDC/MTI	

FOREWORD

This technical report covers work performed under Air Force Contract F33600-87-C-0464, DAPro Project. This contract is sponsored by the Manufacturing Technology Directorate, Air Force Systems Command, Wright-Patterson Air Force Base, Ohio. It was administered under the technical direction of Mr. Bruce A. Rasmussen, Branch Chief, Integration Technology Division, Manufacturing Technology Directorate, through Mr. David L. Judson, Project Manager. The Prime Contractor was Integration Technology Services, Software Programs Division, of the Control Data Corporation, Dayton, Ohio, under the direction of Mr. W. A. Osborne. The DAPro Project Manager for Control Data Corporation was Mr. Jimmy P. Maxwell.

The DAPro project was created to continue the development, test, and demonstration of the Integrated Information Support System (IISS). The IISS technology work comprises enhancements to IISS software and the establishment and operation of IISS test bed hardware and communications for developers and users.

The following list names the Control Data Corporation subcontractors and their contributing activities:

<u>SUBCONTRACTOR</u>	<u>ROLE</u>
Control Data Corporation	Responsible for the overall Common Data Model design development and implementation, IISS integration and test, and technology transfer of IISS.
D. Appleton Company	Responsible for providing software information services for the Common Data Model and IDEF1X integration methodology.
ONTEK	Responsible for defining and testing a representative integrated system base in Artificial Intelligence techniques to establish fitness for use.
Simpact Corporation	Responsible for Communication development.
Structural Dynamics Research Corporation	Responsible for User Interfaces, Virtual Terminal Interface, and Network Transaction Manager design, development, implementation, and support.
Arizona State University	Responsible for test bed operations and support.

TABLE OF CONTENTS

	<u>Page</u>		
SECTION 1	SCOPE	1-1	
1.1	Identification	1-1	
1.2	Functional Summary	1-1	
SECTION 2	DOCUMENTS	2-1	
2.1	Reference Documents	2-1	
2.2	Terms and Abbreviations	2-2	
SECTION 3	REQUIREMENTS	3-1	
3.1	Computer Program Definition	3-1	
3.1.1	System Capacities	3-1	
3.1.2	Interface Requirements	3-1	
3.1.2.1	Interface Block Diagram	3-2	
3.1.2.2	Detailed Interface Definition	3-2	
3.1.2.2.1	Physical Terminal	3-3	
3.1.2.2.2	User Interface	3-3	
3.1.2.2.2.1	Master Mode Device Driver		
	Messages	3-3	
3.1.2.2.2.2	Slave Mode Device Driver		
	Messages	3-4	
3.1.2.2.3	Terminal User	3-4	
3.1.2.2.4	Application Program	3-4	
3.1.2.2.5	Virtual Terminal Protocol	3-5	
3.2	Detailed Functional Requirements	3-6	
3.2.1	Application Interface Routines	3-6	
3.2.1.1	Initialize Virtual Terminal	3-7	
3.2.1.2	Get Virtual Terminal Data	3-7	
3.2.1.3	Put Virtual Terminal Data	3-7	
3.2.1.4	Terminate Virtual Terminal	3-8	
3.2.2	Virtual Terminal Monitor	3-8	
3.2.2.1	Initialize Lower Levels	3-8	
3.2.2.2	Terminate Lower Levels	3-8	
3.2.2.3	Put Data to Lower Levels	3-9	
3.2.2.4	Get Data from Lower Levels	3-9	
3.2.2.5	Check for Terminal Input	3-9	
3.2.3	Class Routines	3-10	
3.2.3.1	Initialize Terminal	3-10	
3.2.3.2	Terminate Terminal	3-10	
3.2.3.3	Get Terminal Command	3-11	
3.2.3.4	Put Terminal Command	3-11	
3.2.3.5	Flush Terminal Buffer	3-11	
3.2.4	Device Routines	3-11	
3.2.5	System Routines	3-12	
3.3	Performance Requirements	3-12	
3.3.1	Programming Methods	3-12	
3.3.2	Program Organization	3-12	
3.3.3	Modification Considerations	3-12	
3.3.4	Special Features	3-12	
3.3.5	Expansibility	3-13	
3.4	Adaptation Requirements	3-13	

SECTION 4	QUALITY ASSURANCE PROVISIONS	4-1
4.1	Introduction and Definitions	4-1
4.2	Computer Programming Test and Evaluation .	4-1
SECTION 5	PREPARATION FOR DELIVERY	5-1
APPENDIX A	VIRTUAL TERMINAL COMMAND DESCRIPTIONS	A-1
APPENDIX B	CHARACTER SET MAPPINGS	B-1

LIST OF ILLUSTRATIONS

<u>Figure</u>	<u>Title</u>	<u>Page</u>
3-1	Virtual Terminal Interfaces	3-2
3-2	Sample COBOL Program Using Direct Interface	3-5

Accession For	
NTIS GRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input checked="" type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	



SECTION 1

SCOPE

1.1 Identification

This specification establishes the performance, development, test, and qualification requirements of the Virtual Terminal protocol and the computer programs implementing it (the Virtual Terminal software). The Virtual Terminal is one configuration item of the Integrated Information Support System User Interface.

1.2 Functional Summary

One of the objectives of the Integrated Information Support System is to allow applications to be run from a wide variety of terminals. Instead of having to worry about device specific commands to perform a specific function on a particular type of terminal, an application instead uses the Virtual Terminal protocol. The Virtual Terminal protocol is defined in terms of the set of functions which it can perform, the set of attributes that it supports, and the commands for invoking these functions.

The Virtual Terminal software translates between the Virtual Terminal commands and commands for the particular type of terminal a user has. Since most terminals do not provide all of the functions and attributes that the Virtual Terminal does, the Virtual Terminal software must be able to simulate missing functions using only existing ones.

In addition to supporting terminals, the Virtual Terminal software also performs another function -- interfacing existing applications into IISS. An existing application sends (and expects to receive) commands for a particular type of terminal. In IISS these commands are intercepted and sent to the Virtual Terminal software which then converts the commands into Virtual Terminal commands, just as if they had been entered from a terminal. Similarly, it converts Virtual Terminal commands to the specific terminal commands the application expects to receive. This allows an existing application to be run from a type of terminal other than the one it was designed for.

SECTION 2
DOCUMENTS

2.1 Reference Documents

- [1] American National Standards Institute, Coded Character Sets - 7-Bit American National Standard Code for Information Interchange (7-Bit ASCII), ANSI X3.4-1986, 26 March 1986.
- [2] American National Standards Institute, Code Extension Techniques for Use with the 7-Bit Coded Character Set of American National Standard Code for Information Interchange, ANSI X3.41-1974, 14 May 1974.
- [3] American National Standards Institute, Additional Controls for Use with American National Standard Code for Information Interchange, ANSI X3.64-1979, 18 July 1979.
- [4] American National Standards Institute, Hollerith Punched Card Code, ANSI X3.26-1980, 2 May 1980.
- [5] International Organization for Standardization, Information Processing - ISO 7-Bit Character Set for Information Interchange, ISO 646-1983.
- [6] International Organization for Standardization, ISO 7-Bit and 8-Bit Coded Character Sets - Code Extension Techniques, ISO 2022-1982.
- [7] International Organization for Standardization, ISO 7-Bit and 8-Bit Coded Character Sets - Additional Control Functions for Character - Imaging Devices, ISO 6429-1983.
- [8] ICAM Documentation Standards, IDS150120000C, 15 September 1983.
- [9] Structural Dynamics Research Corporation, IISS Terminal Operator Guide, OM 620144000, 1 November 1985.
- [10] Structural Dynamics Research Corporation, Form Processor Development Specification, DS 620144200B, 1 November 1985.
- [11] Structural Dynamics Research Corporation, IISS Form Processor User Manual, UM 620144200B, 1 November 1985.
- [12] Structural Dynamics Research Corporation, Virtual Terminal User Manual, UM 620144300B, 1 November 1985.
- [13] Structural Dynamics Research Corporation, Application Interface Development Specification, DS 620144700, 1 November 1985.
- [14] Boeing Military Airplane Company, NTM Programmer's Manual, PRM620242000, 20 July 1987.

2.2 Terms and Abbreviations

AI: Application Interface.

American Standard Code for Information Interchange: the character set defined by ANSI X3.4 and used by most computer vendors.

AP: Application Process.

Application Interface: subset of the IISS User Interface that consists of the callable routines that are linked with applications that use the Form Processor or Virtual Terminal. The Application Interface enables applications to be hosted on computers other than the host of the User Interface.

Application Process: a cohesive unit of software that can be initiated as a unit to perform some function or functions.

ASCII: American Standard Code for Information Interchange.

Attention Interrupt: an asynchronous signal usually generated by pressing a special key or combination of keys which indicates that the currently executing program should be stopped.

Attribute: visual and logical characteristics (e.g. blinking, highlighted, enterable).

Block Mode: a mode of interaction in which a device does not send characters as they are entered but waits for some triggering event to send an entire block of characters.

Character Set: the characters used by a computer system and the bit patterns assigned to represent them.

Class Routines: routines which are common to a number of systems or devices.

Computer Program Configuration Item: an aggregation of computer programs or any of their discrete portions, which satisfies an end-use function.

Control Character: a control function which is coded as a single character.

Control Function: an action that affects the recording, processing, transmission, or interpretation of data.

Control Sequence: a sequence of characters beginning with the Control Sequence Introducer control function which represents a control function.

Control String: a sequence of characters beginning and ending with a delimiting control function which represents a control function.

CPCI: Computer Program Configuration Item.

DD: Device Driver.

Device Driver: software which handles input and output for a specific kind of terminal by mapping between terminal specific commands and Virtual Terminal commands.

Device Routines: routines which are specific to a particular type of terminal.

EBCDIC: Extended Binary Coded Decimal Interchange Code.

Escape Sequence: a sequence of characters beginning with the Escape control character which represent a control function.

Extended Binary Coded Decimal Interchange Code: the character set used by a few computer vendors (notably IBM) instead of ASCII.

Field: two dimensional space on a terminal screen which contains data.

Fill Area: a polygonal area the interior of which is filled-in in some fashion.

Host: a computer system running the Integrated Information Support System.

IISS: Integrated Information Support System.

Integrated Information Support System: a test computing environment used to investigate, demonstrate and test the concepts of information management and information integration in the context of Aerospace Manufacturing. IISS addresses the problems of integration of data resident on heterogeneous databases supported by heterogeneous computers interconnected via a local area network.

Line: a group of points which are connected in the order specified.

Marker: a group of points the locations of which are indicated by a symbol.

Master Mode: the mode of operation used by a Device Driver which is started by a user using host operating system commands in order to connect into IISS.

Network Transaction Manager: IISS subsystem that performs the coordination, communication, and housekeeping functions required to integrate the Application Processes and System Services resident on the various hosts into a cohesive system.

NTM: Network Transaction Manager.

Numeric Parameter: a parameter which represents a numeric value.

Operating System: software supplied with a computer which allows it to supervise its own operations and manage access to hardware facilities such as memory and peripherals.

Physical Device: a hardware terminal.

Primitive: a graphic element contained in a view. Either a fill area, line, marker, or text.

Script File: a file containing Virtual Terminal commands which can be replayed to duplicate the session during which it was recorded.

Selective Parameter: a parameter which is used to represent selections from a list of possible values.

Slave Mode: the mode of operation used by a Device Driver which is started by the User Interface.

Software Control String: a control string which is intended to be used to control software as opposed to hardware.

System Routines: routines which are specific to a particular computer system or operating system.

Text: a primitive consisting sequence of characters the size, color, orientation, and location of which may be specified.

UI: User Interface.

UIM: User Interface Monitor.

UIMS: User Interface Management System.

User Interface: IISS subsystem that controls the user's terminal and interfaces with the rest of the system. The UI consists of two major subsystems: the User Interface Development System and the User Interface Management System.

View: an area of the screen which can contain primitives.

Virtual Terminal: subset of the IISS User Interface that performs the interfacing between different terminals and the UI. This is done by defining a specific set of terminal features and protocols which must be supported by the UI software which constitutes the virtual terminal definition. Specific terminals are then mapped against the virtual terminal software by specific software modules written for each type of real terminal supported.

Virtual Terminal Interface: the callable interface to the Virtual Terminal.

VT: Virtual Terminal.

VTI: Virtual Terminal Interface.

Window: an area of the screen which can contain other windows (sub-windows), fields, and views.

Window Manager: a facility which allows the following to be manipulated: size and location of windows, the device on which an application is running, the position of a form within a window. It is part of the Form Processor.

SECTION 3

REQUIREMENTS

3.1 Computer Program Definition

The Virtual Terminal protocol and software allow Integrated Information Support System applications to run on a wide variety of physical devices (terminals). The Virtual Terminal protocol defines the available functions and attributes and the Virtual Terminal software translates between the Virtual Terminal protocol and commands for a particular type of terminal. An implementation of the Virtual Terminal software for a specific operating system and terminal type is referred to as a Device Driver.

3.1.1 System Capacities

A Device Driver is required to handle only a single terminal as there will be a separate instance of the software for each active terminal. Dynamic memory allocation will be used to avoid any artificial limitations on screen size or image complexity.

3.1.2 Interface Requirements

A Device Driver has direct interfaces with the Network Transaction Manager and the host operating system. It has indirect interfaces with the User Interface, a physical terminal, application programs, and the user of the terminal.

The interface between a Device Driver and the User Interface is provided by Network Transaction Manager messages which are sent and received using the standard NTM Services defined in [14]. The interface between a Device Driver and application programs is also provided by NTM messages which are initially generated by Virtual Terminal Services which are part of the Application Interface [13]. These messages are routed through the User Interface which reformats them as required before sending them on to their ultimate destination.

The content of these messages is specified by the Virtual Terminal protocol as defined in section 3.1.2.2.5. The protocol is to conform to existing national and international standards [1], [2], [3], [4], [5], [6], and [7]. Although these standards are defined in terms of a standard character set, the Virtual Terminal protocol will be represented using the native character set of the host machine (character set conversion is provided by the Network Transaction Manager). Character set mappings for the most common character sets are provided in Appendix B.

The interface between the Virtual Terminal software and a physical terminal is provided by the host operating system. The actual mechanism for sending and receiving terminal data is thus system specific, and the format of the data is device specific.

3.1.2.1 Interface Block Diagram

The Virtual Terminal structure and interfaces are depicted graphically in Figure 3-1. Boxes represent processes and arrows represent NTM messages. Divisions within a box indicate separate subsystems.

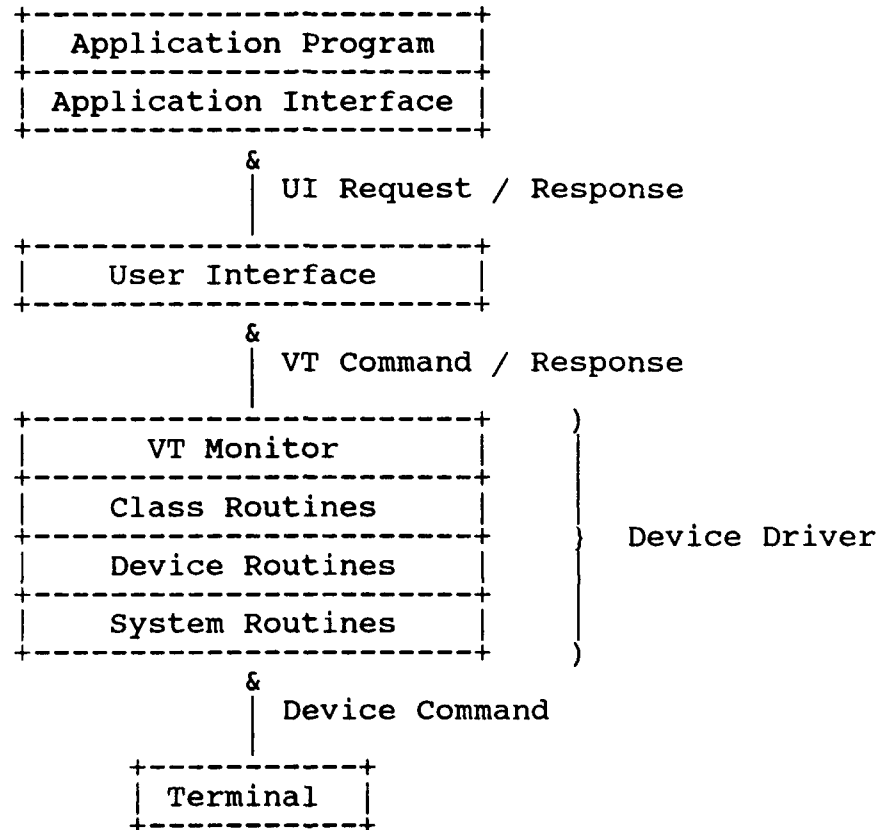


Figure 3-1 Virtual Terminal Interfaces

3.1.2.2 Detailed Interface Definition

Device Drivers are operated in one of two modes: master and slave. Master mode is used when a Device Driver is initiated by a user in order to connect to the Integrated Information Support System. This is done using host operating system commands and results in the Device Driver being connected to the user's terminal. Slave mode is used when a Device Driver is initiated by the User Interface in response to a connected user's request to use an additional device. This is done using an NTM message and results in the Device Driver being connected to the device specified in the message. These modes will not be mentioned except where there are differences between them.

3.1.2.2.1 Physical Terminal

As noted before, the interface to a physical terminal is provided by host operating system services and is thus both system and device dependent.

3.1.2.2.2 User Interface

The interface to the User Interface is provided by NTM messages the types and interpretations of which are defined in the following sections. The Virtual Terminal commands contained in these messages are defined by the Virtual Terminal protocol in section 3.1.2.2.5. This interface is a block mode interface; characters are received from the terminal in a block rather than one at a time. Transmission is triggered when the user presses a function key.

3.1.2.2.2.1 Master Mode Device Driver Messages

Received Messages

SD	Shutdown - Terminate Device Driver.
DD	Device Data - Process Virtual Terminal commands contained in message.
DQ	Device Query - Process Virtual Terminal commands contained in message and respond with an acknowledgment when finished.
SE	Signal Error - Forward message to User Interface Monitor.
UL	User Logon - Perform NTM Logon with parameters contained in message.
UC	User Change Role - Perform NTM Change Role with parameters contained in message.

Sent Messages

DD	Device Data - Sent when the user presses a function key - Contains Virtual Terminal commands for user entered data.
DA	Device Acknowledgment - Sent in response to a Device Query message when processing is complete.
ER	Error Report - Sent when an error is detected - Contains information about the error.
DE	Device Enable - Sent during initialization - Specifies the characteristics of the newly started Virtual Terminal.
AB	Abort - Sent when an unrecoverable error is detected - Contains information about the error.
FE	Forwarded Error - Sent in response to a Signal Error message - Contains the contents of the Signal Error message.

3.1.2.2.2 Slave Mode Device Driver Messages

Received Messages

DE Device Enable - Initialize Virtual Terminal for device contained in message.
SD Shutdown - Terminate Virtual Terminal.
DD Device Data - Process Virtual Terminal commands contained in message.

Sent Messages

ER Error Report - Sent when an error is detected - Contains information about the error.
DI Device Initiation - Sent during initialization - Specifies the characteristics of the newly started Virtual Terminal.
AB Abort - Sent when an unrecoverable error is detected - Contains information about the error.

3.1.2.2.3 Terminal User

When invoking a master mode Device Driver, the user can specify a number of parameters on the command line to enable optional processing. Available options include reading and writing script files, saving output, and screen printing. Scripting is actually performed by the User Interface; the Device Driver is responsible only for requesting it. The options are specified as follows:

-w <script file name> - write script file
-a <script file name> - append to existing script file
-r <script file name> - read script file
-s <save file name> - saves output from session
-p - print every screen

The -r, -w, and -a options are mutually exclusive.

3.1.2.2.4 Application Program

Application programs normally interface with the User Interface rather than the Virtual Terminal, but a set of Application Interface routines are provided for special situations where a direct interface is required. An application program using this direct interface must call the Virtual Terminal initialization (INITVT) and termination (TERMVT) routines to bypass the Form Processor. Since INITVT and TERMVT merely toggle Form Processor bypass mode, INITFP and TERMFP must be still be called at the beginning and end of the application as illustrated by the sample COBOL program in Figure 2-3.

3.1.2.2.5 Virtual Terminal Protocol

The Virtual Terminal Protocol consists of the control functions specified in Appendix A. These control functions are defined in terms of the ASCII character set but are actually represented in the local host character set. Mappings of the most common character sets are specified in Appendix B.

Each control function is represented in one of three ways: as a control character, an escape sequence, or a control string. The control characters are those from 0 to 31 and 127; their meanings are as specified by [1] and [5]. The general form of escape sequences is given in [2] and [6] with specific meanings assigned in [3] and [7]. Control strings are also defined in [3] and [7]. In the remainder of this document control functions will be referred to by name or by their standard acronyms enclosed in angle brackets (e.g. <CR>). Characters will be referred to by name, acronym, and/or decimal value as appropriate.

```

APPLICATION PROGRAM

.
.
.
PROCEDURE DIVISION.
  CALL "INITFP".
  .
  .
  .
  CALL "INITVT".
  CALL "PUTVTI" USING COMMANDS, COMMAND-LEN.
  CALL "GETVTI" USING BUFFER, BUF-SIZE, BUF-LEN.
  .
  .
  .
  CALL "TERMVT".
  .
  .
  .
  CALL "INITVT".
  .
  .
  .
  CALL "TERMVT".
  .
  .
  .
  CALL "TERMF".

```

Figure 3-2 Sample COBOL Program Using Direct Interface

The general format of an escape sequence is the <ESC> character (27), zero or more intermediate characters in the range 32 to 47, and a final character in the range 48 to 126. The general format of a control string is an opening delimiter, a string of graphic characters and spaces (the range 32 to 126), and a closing delimiter. The control strings fall into two classes: software control strings and control sequence functions. The software control string begins with <APC> and ends with <ST>.

The general format of a control sequence function is the <CSI> escape sequence, a parameter string consisting of zero or more characters in the range 48 to 63, zero or more intermediate characters in the range 32 to 47, and a final character in the range 64 to 126. A parameter string consists of zero or more parameters separated by semicolons (59). Each parameter consists of characters in the range 48 to 57 which are interpreted as the digits 0 through 9. Parameters are interpreted as decimal integers; leading zeros are not significant and may be omitted. A zero length parameter or one consisting only of zeros represents a default value. Trailing zero parameters may be omitted, as may their separators.

There are two types of parameters: numeric parameters are used for passing numeric values, selective parameters are used to select particular entries from a list. The form of both types is the same. Although there are no standardized functions which require both numeric and selective parameters, the colon (58) has been reserved for use as the separator between numeric and selective parameters.

3.2 Detailed Functional Requirements

In order to allow the code to be as portable and reusable as possible, functions are separated into Class Routines, Device Routines, and System Routines. Devices Routines are those that are applicable to a particular type of terminal, regardless of the host operating system. System Routines are those that are applicable to a particular operating system, regardless of the type of terminal. Class Routines are those that are applicable to a large class of terminals and operating systems (e.g. the routines that support window management on non-windowing terminals and operating systems).

3.2.1 Application Interface Routines

These routines are responsible for initiating, terminating, and communicating with a Device Driver being run as a separate process; they form the Application Program side of the communication link to the Device Driver.

3.2.1.1 Initialize Virtual Terminal

Initialize Virtual Terminal (INITVT) performs all necessary initialization in preparation for using the Virtual Terminal and puts the Form Processor in bypass mode.

Calling Sequence

CALL "INITVT".

3.2.1.2 Get Virtual Terminal Data

Get Virtual Terminal Data (GETVTI) accepts user input, converts that input from a device specific form to a neutral form, and returns the neutral form data to the caller. All user input is accumulated until a function key is pressed. The returned data consists of a formatted buffer of the screen contents. The buffer consists of a Set Window command followed by Define Field commands for each field in the window which has been modified since the last read. This is followed by additional Set Window and Define Field commands for nested windows. Finally, a Cursor Position Report command giving the cursor position when the terminating function key was pressed and an Application Program Command command specifying which function key was pressed terminate the buffer.

Calling Sequence

CALL "GETVTI" USING BUFFER, MAX-LEN, ACT-LEN.

Inputs

MAX-LEN - binary - Maximum length to read.

Outputs

BUFFER - character - Data read from terminal.
ACT-LEN - binary - Length of data read.

3.2.1.3 Put Virtual Terminal Data

Put Virtual Terminal Data (PUTVTI) converts the supplied neutral form data to device specific data and displays it to the user. The supplied data may contain any of the Virtual Terminal commands.

Calling Sequence

CALL "PUTVTI" USING BUFFER, ACT-LEN.

Inputs

BUFFER - character - Data to be written.
ACT-LEN - binary - Length of data to write.

3.2.1.4 Terminate Virtual Terminal

Terminate Virtual Terminal (TERMVT) performs all necessary clean-up operations and terminates use of the Virtual Terminal by disabling Form Processor bypass mode.

Calling Sequence

CALL "TERMVT".

3.2.2 Virtual Terminal Monitor

This module is responsible for message processing. Messages received from the User Interface are processed as specified in section 3.1.2.2.2. If a message contains Virtual Terminal commands, they are passed on to lower level modules for device and system specific processing. The lower level routines called by the monitor are described in the following sections.

3.2.2.1 Initialize Lower Levels

This routine is called to allow the lower level routines to perform any necessary initialization and to get information about the size and capabilities of the terminal.

Calling Sequence

CALL "INTVT" USING NAME, LEN, SIZ, GRF, SURF, COLORS.

Inputs

NAME - Terminal Name (ignored in master mode)
LEN - Length of NAME (0 if null terminated)

Outputs

SIZE - Width and Depth of Terminal in Character Cells
GRF - Terminal Supports Graphics (0 or 1)
SURF - Width and Depth of Terminal in Graphic Pixels
COLORS - Number of Colors Terminal Supports

Returns

Zero for failure, nonzero for success.

3.2.2.2 Terminate Lower Levels

This routine is called to allow the lower level routines to perform any necessary cleanup prior to termination.

Calling Sequence

CALL "TRMVT".

3.2.2.3 Put Data to Lower Levels

This routine is called to send Virtual Terminal commands to the lower level routines for processing.

Calling Sequence

CALL "PUTVT" USING BUFFER, LEN.

Inputs

BUFFER - Buffer of Virtual Terminal Commands
LEN - Length of BUFFER

Returns

Zero for failure, nonzero for success.

3.2.2.4 Get Data from Lower Levels

This routine is called to allow the lower level routines to process user input and to retrieve any resulting Virtual Terminal commands.

Calling Sequence

CALL "GETVT" USING BUFFER, SIZE.

Inputs

SIZE - Size of BUFFER

Outputs

BUFFER - Buffer of Virtual Terminal Commands

Returns

Length actually used in BUFFER.

3.2.2.5 Check for Terminal Input

This routine is called to determine if GETVT should be called to process input.

Calling Sequence

CALL "TRMCHK".

Returns

Number of input characters pending (zero / nonzero is sufficient).

3.2.3 Class Routines

Class Routines are those which are applicable to a variety of systems and devices. By convention, these routines are given names ending with "VT". Class Routines include the window manager for systems and devices which do not have windowing capabilities and the block mode simulator.

The window manager maintains a data structure representing the hierarchy of windows, views, fields, and primitives which is defined and modified by Virtual Terminal commands. The final display image is determined by traversing this data structure and determining the visibility of each element. Similarly, the data structure is traversed when an input request is satisfied in order to build the appropriate sequence of Virtual Terminal commands to be returned.

The block mode emulator is an adjunct to the window manager which allows an interactive terminal to work like a field structured block mode terminal. The emulator gets commands from the Device Routines and interprets them. Cursor positioning commands are examined in order to keep track of the current position in the window manager data structure. Printable characters are inserted into the structure and displayed on the screen at the correct location.

The window manager and block mode emulator provide the INIVT, TRMVT, GETVT, and PUTVT routines required by the monitor. They reference the Device Routines described in the following sections.

3.2.3.1 Initialize Terminal

This routine is called to allow the Device Routines to perform any necessary initialization.

Calling Sequence

CALL "TRMINI" USING NAME.

Inputs

NAME - Terminal Name (null terminated)

Returns

Zero for failure, nonzero for success.

3.2.3.2 Terminate Terminal

This is called to allow the Device Routines to perform any necessary cleanup prior to termination.

Calling Sequence

CALL "TRMEND".

3.2.3.3 Get Terminal Command

This routine is called to get a Virtual Terminal command from the Device Routines.

Calling Sequence

CALL "TRMGET" USING CMD.

Outputs

CMD - Virtual Terminal Command

Returns

Zero for failure, nonzero for success.

3.2.3.4 Put Terminal Command

This routine is called to pass a Virtual Terminal command to the Device Routines for execution.

Calling Sequence

CALL "TRMPUT" USING CMD.

Inputs

CMD - Virtual Terminal Command

3.2.3.5 Flush Terminal Buffer

This routine is called to flush any buffers the Device Routines may have and complete any deferred actions to ensure that the screen image is up to date.

Calling Sequence

CALL "TRMFLS".

3.2.4 Device Routines

Device Routines are responsible for mapping between device specific commands and Virtual Terminal commands. By convention, these routines are given names beginning with "TRM" and all routines for a single terminal reside in the same file.

3.2.5 System Routines

System Routines are used by the Device Routines to communicate with a physical terminal. For block mode emulation, the system routines must provide a mechanism for reading individual characters as they are entered without echoing the characters to the screen.

3.3 Performance Requirements

3.3.1 Programming Methods

The Virtual Terminal is implemented as a number of Device Drivers; the correct Device Driver for a particular terminal is selected at run-time. Dividing the routines into Device, System, and Class routines allows for sharing much of the actual Device Driver code. Different devices on the same system can share the System and Class routines. The same device on different systems can share Device and Class routines. Similar systems share Class routines. This system of sharable and reusable code simplifies the process of implementing new device drivers.

All Class routines will be written in portable C for ease of porting to additional systems. Device routines will also be written in portable C with the possible exception of devices which are intimately tied to a particular system. System routines will be written in the language most suited to interacting with the host operating system.

3.3.2 Program Organization

The program organization is as specified in section 3.2.

3.3.3 Modification Consideration

Modification considerations were a major consideration in the decision to use the structure specified in section 3.2. Isolating Device Routines allows installation specific changes (e.g. alternate keyboard mappings) to be made conveniently since the affected module is easily located. Further modification such as new devices or systems is also facilitated by the large number of reusable routines.

3.3.4 Special Features

The design is such that the monitor may be easily replaced during testing with an equivalent module which allows the lower level routines to be linked directly to the User Interface. This aids in fault isolation and diagnosis by eliminating the dependency upon the Network Transaction Manager.

3.3.5 Expansibility

Since the Virtual Terminal is implemented as Device Drivers, it may be expanded to support any number of device types by implementing new Device Drivers. Separating functions into Device, System, and Class routines aids in this process by insuring that reusable routines can be easily identified.

3.4 Adaptation Requirements

Adapting the Virtual Terminal for use on other systems requires modification to or replacement of the System Routines. Adaptation for use with other terminal requires creation of new Device Routines.

SECTION 4

QUALITY ASSURANCE PROVISIONS

4.1 Introduction and Definitions

Testing is a systematic process that may be preplanned and explicitly stated. Test techniques and procedures may be defined in advance and a sequence of test steps may be specified. Debugging is the process of isolation and correction of the cause of an error.

Antibugging is defined as the philosophy of writing programs in such a way as to make bugs less likely to occur and when they do occur, to make them more noticeable to the programmer and the user. In other words, as much error checking as is practical and possible in each routine should be performed.

4.2 Computer Programming Test and Evaluation

The quality assurance provisions for test consist of the normal testing techniques that are accomplished during the construction process. They consist of design and code walk-throughs, unit testing, and integration testing. These tests are performed by the design team. Structured design, design walk-through and the incorporation of antibugging facilitate this testing by exposing and addressing problem areas before they become coded bugs.

The integration testing entails use of a test application of the VAX. This test program will display forms, read input from forms, and display results.

Each function is be tested separately, then the entire subsystem is tested as a unit. All testing is done on the IISS testbed.

SECTION 5

PREPARATION FOR DELIVERY

The implementation site for the constructed software is the Integrated Support System Test Bed site provided by the Air Force. The software associated with each CPCI release is delivered on a media which is compatible with the IISS Test Bed. The release is clearly identified and includes instructions on procedures to be followed for installation of the release. Integration with the other IISS CPCI's is done on the IISS testbed on a scheduled basis.

APPENDIX A

VIRTUAL TERMINAL COMMAND DESCRIPTIONS

The format of the following command descriptions is the control function acronym and name, the command syntax, and a detailed description of the command. In the command syntax, characters within angle brackets (e.g. <ESC>) indicate other control functions, Pn indicates a Numeric Parameter, Ps indicates a Selective Parameter, and all other characters stand for themselves. Unless specified otherwise, Numeric Parameters indicate the number of times to repeat the specified function, omitted Numeric Parameters are taken to be 1, and omitted Selective Parameters are taken to be 0.

The Virtual Terminal screen consists of X rows numbered from 1 to X, and Y columns numbered from 1 to Y. The standard ordering of objects is from top to bottom and left to right, with wrap-around from the last object to the first. "Next" in the command descriptions refers to this order, "previous" to its reverse. For example (if Y=80), from row 6 column 80, the next character position is row 7 column 1, and the previous character position is row 6 column 79.

Any command whose effect is limited to a single field (including Graphic Characters) causes the cursor to move to the next unprotected field before the command takes effect if the cursor is in a protected field when the command is received. If there are no unprotected fields defined, the command is ignored.

Each window is assigned a coordinate system ranging from 0 to 65535 in both X and Y. This coordinate system is used only for setting the window's viewport. Each viewport is also assigned a coordinate system ranging from 0 to 65535 in both X and Y. All graphic primitives are specified in viewport coordinates.

Control Characters

BEL - Sound Bell

<BEL>

Sounds an audible alarm at the terminal.

BS - Backspace

<BS>

Moves the cursor to the previous character position. If the cursor is at the first column, it is moved to the last column of the previous line or the bottom line if the cursor was on the top line.

HT - Horizontal Tab

<HT>

Moves the cursor to the next field.

LF - Line Feed
<LF>

Moves the cursor down to the next line in the current column. If the cursor is at the bottom line of the screen, it is moved to the next column in the top line or the first column in the cursor was in the last column.

FF - Form Feed
<FF>

Erases all unprotected fields on the screen and moves the cursor to the top left corner of the screen.

CR - Carriage Return
<CR>

Moves the cursor backward to the beginning of the previous field.

RS - Record Separator
<RS>

Causes the screen to be updated. To improve efficiency, commands are not necessarily executed when received. This command is given at the end of a series of commands to force any pending actions to be completed.

US - Unit Separator
<US>

Causes the screen to be remapped. To improve efficiency, the mapping between fields and characters on the screen is not updated with every change. This command is given at the end of a series of format changes to enable remapping.

SP - Space
<SP>

Treated as a graphic character.

Graphic Characters

Causes the specified character to be displayed according to the field attributes in effect at the cursor location and advances the cursor to the next character position.

Extended Control Characters / Escape Sequences

REF - Refresh Screen
<ESC> ?

Retransmits the current screen contents to the terminal. Its main uses are to recover from unsolicited messages or line noise which have corrupted the screen contents.

IND - Index
<IND>

<ESC> D

Same as Line Feed.

NEL - Next Line

<NEL>
<ESC> E

Moves the cursor to the beginning of the next line of the current field or the first line of the next field if the cursor is on the last line of a field.

RI - Reverse Index

<RI>
<ESC> M

Moves the cursor up to the previous line in the current column. If the cursor is on the top line of the screen it is moved to the last line in the previous column or the last column if the cursor was in the first column.

CEL - Cursor End of Line

<ESC> Q

Moves the cursor to the end of the next line of the current field or the first line of the next field if the cursor is at the end of the last line of a field.

STS - Set Transmit State

<STS>
<ESC> S

Enables the currently selected window and all its contained subwindows for input. All unguarded fields are made enterable and a data message will be sent when a function key is pressed. The data message contains a Select Window command for each window which has been modified. Each Define Window command is followed by Define Field commands for each contained modified field (including the field data) and Define Window commands for each contained modified window. Following all of these commands is a Cursor Position Report command giving the cursor position when the function key was pressed and an Application Program Command specifying the function key.

CSI - Control Sequence Introducer

<CSI>
<ESC> [

Indicates the beginning of a Control Sequence.

ST - String Terminator

<ST>
<ESC> \

Terminates an Application Program Command.

APC - Application Program Command

<APC>
<ESC>

Generated when a function key is pressed. This command is followed by the function key number (0 - N) and a String Terminator. Function key zero is the "ENTER" key.

RIS - Reset to Initial State

<ESC> c

Resets the terminal to its initial state. The screen is cleared, the cursor is positioned in the upper left corner.

Control Sequences

ICH - Insert Character

<CSI> Pn @

Makes room for a character by shifting the current and following characters of the field one character position to the right; characters shifted past the end of the field are lost. The cursor is left at the first inserted character position (i.e. not moved).

CUU - Cursor Up

<CSI> Pn A

Same as Reverse Index.

CUD - Cursor Down

<CSI> Pn B

Same as Line Feed.

CUF - Cursor Forward

<CSI> Pn C

Moves the cursor to the next character position. If the cursor is in the last column it is moved to the first column of the next line or the first line if the cursor was in the last line.

CUB - Cursor Backward

<CSI> Pn D

Moves the cursor to the previous character position. If the cursor is in the first column it is moved to the last column of the previous line or the last line if the cursor was on the top line.

CNL - Cursor Next Line

<CSI> Pn E

Same as Next Line.

CPL - Cursor Previous Line

<CSI> Pn F

Moves the cursor to the beginning of the previous line of the field or the previous field if the cursor was at the beginning of a field.

CUP - Cursor Position

<CSI> Pn ; Pn H

Moves the cursor to the specified position. The first parameter is the row number, the second parameter is the column number.

CHT - Cursor Horizontal Tab

<CSI> Pn I

Same as Horizontal Tab.

ED - Erase Display

<CSI> Ps J

Erases unprotected fields on the screen according to the parameter:

0 - Erase from the cursor to the end of the screen

- 1 - Erase from the beginning of the screen to the cursor
- 2 - Erase the entire screen

The cursor is not moved.

EL - Erase Line

<CSI> Ps K

Erases unprotected fields on the current line according to the parameter:

- 0 - Erase from the cursor to the end of the line
- 1 - Erase from the beginning of the line to the cursor
- 2 - Erase the entire line

The cursor is not moved.

IL - Insert Line

<CSI> Pn L

Makes room for a line by shifting the current and following line of the field down one line; lines shifted past the bottom of the field are lost. The cursor is positioned at the first inserted line (i.e. not moved).

DL - Delete Line

<CSI> Pn M

Deletes the current line by shifting the following lines of the field up one line.

EF - Erase Field

<CSI> Ps N

Erases the current field according to the parameter:

- 0 - Erase from the cursor to the end of the field
- 1 - Erase from the beginning of the field to the cursor
- 2 - Erase the entire field

The cursor is not moved.

DCH - Delete Character

<CSI> Pn P

Deletes the current character by shifting the following characters of the field one character position to the left.

CPR - Cursor Position Report

<CSI> Pn ; Pn R

Indicates the current position of the cursor.

NR - Next Page

<CSI> Pn U

Same as Form Feed.

PP - Previous Page

<CSI> Pn V

Same as Form Feed.

ECH - Erase Character

<CSI> Pn X

Erases the specified number of characters starting at the current character (the characters are NOT deleted). The cursor is not moved.

CBT - Cursor Backward Tab

<CSI> Pn Z

Moves the cursor to the previous field.

HPA - Horizontal Position Absolute

<CSI> Pn

Moves the cursor to the specified column in the current line. If the specified position is beyond the last column the cursor is moved to the first column of the next line or the first line if the cursor was in the last line.

HPR - Horizontal Position Relative

<CSI> Pn a

Same as Cursor Forward.

VPA - Vertical Position Absolute

<CSI> Pn d

Moves the cursor to the specified line in the current column. If the specified position is beyond the last row the cursor is moved to the next column in the first line or the first column if the cursor was in the last column.

VPR - Vertical Position Relative

<CSI> Pn e

Same as Line Feed.

HVP - Horizontal and Vertical Position

<CSI> Pn ; Pn f

Same as Cursor Position.

SM - Set Mode

<CSI> Ps h

Sets the indicated mode:

4 - Insertion - Replacement (Insert)

MC - Media Copy

<CSI>Ps i

Controls the transfer of data between the device and an auxiliary input/output device:

0 - Print Screen

RM - Reset Mode

<CSI> Ps l

Resets the indicated modes:

4 - Insertion - Replacement (Replacement)

WP - Set Window Precedence

<CSI> Pn ; ... p

Sets the precedence of the specified windows. Each is in turn placed on top of all other existing windows. Thus, the last window specified will ultimately be the top-most, and all specified windows will be on top of any unspecified windows.

RW - Remove Window

<CSI> Pn r

Removes the window from the screen.

SW - Select Window

<CSI> Pn s

Selects the specified window. If the parameter is zero or omitted, the device itself is selected.

EW - Erase Window

<CSI> Pn u

Removes all subwindows, fields, and graphic primitives from the specified subwindow of the currently selected window. If the parameter is zero or omitted, the currently selected window itself is erased.

DV - Define Viewport

<CSI> Pn ; Pn ; Pn ; Pn v

Sets the graphic viewport of the currently selected window. The viewport is assigned a virtual coordinate system extending from 0 to 65535 in both X and Y. Graphic primitives are specified in viewport coordinates and thus are completely contained within the viewport at all times. When a window is created, the viewport is set to the entire window (0;0;65535;65535). The numeric parameters are the X and Y window coordinates of the lower left corner of the viewport and the X and Y window coordinates of the upper right corner of the viewport.

DW - Define Window

<CSI> Pn ; Pn ; Pn ; Pn ; Pn ; Pn ; Pn ; Pn ; Pn ; Pn : Ps w

Defines or modifies a subwindow of the currently selected window. The numeric parameters are the window ID, the row within the parent window, the column within the parent window, the displayed width, the displayed depth, the number of rows the display is offset, the number of columns the display is offset, the logical width, and the logical depth. The selective parameter is the attributes for the window:

- 40 - Black Background
- 41 - Red Background
- 42 - Green Background
- 43 - Yellow Background
- 44 - Blue Background
- 45 - Magenta Background
- 46 - Cyan Background
- 47 - White Background
- 30 - Black Foreground
- 31 - Red Foreground
- 32 - Green Foreground
- 33 - Yellow Foreground
- 34 - Blue Foreground
- 35 - Magenta Foreground
- 36 - Cyan Foreground
- 37 - White Foreground

Omitted parameters indicate no change to the existing values. (Note that the foreground color is used to determine whether the window appears in normal or reverse video on monochrome terminals.)

DF - Define Field

<CSI> Pn ; Pn ; Pn ; Pn : Ps ; Ps x

Defines or modifies a field in the currently selected window. The

numeric parameters are the row within the parent window, the column within the parent window, the display width, and the display depth. The first selective parameter is the "guarded" flag which must consist of exactly one selection from the following list:

- 0 - Field is enterable
- 1 - Field is guarded

The second selective parameter is the attributes for the field:

- 0 - Normal (Remove any existing attributes)
- 1 - Bright or Bold
- 2 - Dim
- 4 - Underlined
- 5 - Slow Blink (less than 150 per minute)
- 6 - Fast Blink (more than 150 per minute)
- 7 - Reverse
- 8 - Concealed (not displayed)
- 30 - Black Foreground
- 31 - Red Foreground
- 32 - Green Foreground
- 33 - Yellow Foreground
- 34 - Blue Foreground
- 35 - Magenta Foreground
- 36 - Cyan Foreground
- 37 - White Foreground
- 40 - Black Background
- 41 - Red Background
- 42 - Green Background
- 43 - Yellow Background
- 44 - Blue Background
- 45 - Magenta Background
- 46 - Cyan Background
- 47 - White Background

Omitted parameters indicate no change to the existing values. This command is followed by the data for the field. Trailing blanks in the field data may be omitted.

DVT - Define View Text

<CSI> Pn ; Pn ; Pn ; Pn ; Pn ; Pn ; Pn ; Pn : Ps ; Ps ; Ps ; Ps ; Pn " v

Adds a text primitive to the currently selected view. The parameters are the X and Y viewport coordinates of the reference point, the color index, the font index, the spacing factor, the expansion factor, the height, the angle of the up vector (0 = up, 16384 = left, 32768 = down, 49152 = right), the precision, the path, the vertical alignment, the horizontal alignment, and the length of the text string. Only font 1 is currently supported. Parameters indicated as factors are specified as a percentage of the nominal value (omitted factors are taken to be 100). Valid precisions are:

- 0 - String

Valid paths are:

- 0 - Right
- 1 - Left
- 2 - Up
- 3 - Down

Valid vertical alignments are:

- 0 - Normal

- 1 - Top
- 2 - Cap
- 3 - Half
- 4 - Base
- 5 - Bottom

Valid Horizontal alignments are:

- 0 - Normal
- 1 - Left
- 2 - Center
- 3 - Right

DVF - Define View Fill Area

<CSI> Pn ; Pn ; Pn ; Pn ... & v

Adds a fill area primitive to the currently selected view. The parameters are the color index, interior style index, and X and Y viewport coordinates of the boundary. Valid interior styles are:

- 0 - Hollow
- 1 - Solid
- 4 - Empty

DVM - Define View Markers

<CSI> Pn ; Pn ; Pn ; Pn ; Pn ... * v

Adds a marker primitive to the currently selected view. The parameters are the color index, the marker style, the marker size factor, and the X and Y viewport coordinates of the markers. The maker size factor is specified as a percentage of the nominal value. Valid maker styles are:

- 1 - Point
- 2 - Plus
- 3 - Star
- 4 - Circle
- 5 - Cross

DVL - Define View Lines

<CSI> Pn ; Pn ; Pn ; Pn ; Pn ... - v

Adds a line primitive to the currently selected view. The parameters are the color index, the line style, the line width factor, and the X and Y viewport coordinates of the points to be connected. The line width factor is specified as a percentage of the nominal value. Valid line styles are:

- 1 - Solid
- 2 - Dashed
- 3 - Dotted
- 4 - Dashed Dotted

APPENDIX B
CHARACTER SET MAPPINGS

Char	ASCII			EBCDIC			Char	ASCII			EBCDIC		
	Hex	Oct	Dec	Hex	Oct	Dec		Hex	Oct	Dec	Hex	Oct	Dec
<NUL>	00	000	0	00	000	0	<SP>	20	040	32	40	100	64
<SOH>	01	001	1	01	001	1	!	21	041	33	5A	132	90
<STX>	02	002	2	02	002	2	"	22	042	34	7F	177	127
<ETX>	03	003	3	03	003	3	#	23	043	35	7B	173	123
<EOT>	04	004	4	37	067	55	\$	24	044	36	5B	133	91
<ENQ>	05	005	5	2D	055	45	%	25	045	37	6C	154	108
<ACK>	06	006	6	2E	056	46	&	26	046	38	50	120	80
<BEL>	07	007	7	2F	057	47	'	27	047	39	7D	175	125
<BS>	08	010	8	16	026	22	(28	050	40	4D	115	77
<HT>	09	011	9	05	005	5)	29	051	41	5D	135	93
<LF>	0A	012	10	25	045	37	*	2A	052	42	5C	134	92
<VT>	0B	013	11	0B	013	11	+	2B	053	43	4E	116	78
<FF>	0C	014	12	0C	014	12	,	2C	054	44	6B	153	107
<CR>	0D	015	13	0D	015	13	-	2D	055	45	60	140	96
<SO>	0E	016	14	0E	016	14	.	2E	056	46	4B	113	75
<SI>	0F	017	15	0F	017	15	/	2F	057	47	61	141	97
<DLE>	10	020	16	10	020	16	0	30	060	48	F0	360	240
<DC1>	11	021	17	11	021	17	1	31	061	49	F1	361	241
<DC2>	12	022	18	12	022	18	2	32	062	50	F2	362	242
<DC3>	13	023	19	13	023	19	3	33	063	51	F3	363	243
<DC4>	14	024	20	3C	074	60	4	34	064	52	F4	364	244
<NAK>	15	025	21	3D	075	61	5	35	065	53	F5	365	245
<SYN>	16	026	22	32	062	50	6	36	066	54	F6	366	246
<ETB>	17	027	23	26	046	38	7	37	067	55	F7	367	247
<CAN>	18	030	24	18	030	24	8	38	070	56	F8	370	248
	19	031	25	19	031	25	9	39	071	57	F9	371	249
<SUB>	1A	032	26	3F	077	63	:	3A	072	58	7A	172	122
<ESC>	1B	033	27	27	047	39	;	3B	073	59	5E	136	94
<FS>	1C	034	28	1C	034	28	<	3C	074	60	4C	114	76
<GS>	1D	035	29	1D	035	29	=	3D	075	61	7E	176	126
<RS>	1E	036	30	1E	036	30	>	3E	076	62	6E	156	110
<US>	1F	037	31	1F	037	31	?	3F	077	63	6F	157	111

Char	ASCII			EBCDIC			Char	ASCII			EBCDIC		
	Hex	Oct	Dec	Hex	Oct	Dec		Hex	Oct	Dec	Hex	Oct	Dec
@	40	100	64	7C	174	124	.	60	140	96	79	171	121
A	41	101	65	C1	301	193	a	61	141	97	81	201	129
B	42	102	66	C2	302	194	b	62	142	98	82	202	130
C	43	103	67	C3	303	195	c	63	143	99	83	203	131
D	44	104	68	C4	304	196	d	64	144	100	84	204	132
E	45	105	69	C5	305	197	e	65	145	101	85	205	133
F	46	106	70	C6	306	198	f	66	146	102	86	206	134
G	47	107	71	C7	307	199	g	67	147	103	87	207	135
H	48	110	72	C8	310	200	h	68	150	104	88	210	136
I	49	111	73	C9	311	201	i	69	151	105	89	211	137
J	4A	112	74	D1	321	209	j	6A	152	106	91	221	145
K	4B	113	75	D2	322	210	k	6B	153	107	92	222	146
L	4C	114	76	D3	323	211	l	6C	154	108	93	223	147
M	4D	115	77	D4	324	212	m	6D	155	109	94	224	148
N	4E	116	78	D5	325	213	n	6E	156	110	95	225	149
O	4F	117	79	D6	326	214	o	6F	157	111	96	226	150
P	50	120	80	D7	327	215	p	70	160	112	97	227	151
Q	51	121	81	D8	330	216	q	71	161	113	98	230	152
R	52	122	82	D9	331	217	r	72	162	114	99	231	153
S	53	123	83	E2	342	226	s	73	163	115	A2	242	162
T	54	124	84	E3	343	227	t	74	164	116	A3	243	163
U	55	125	85	E4	344	228	u	75	165	117	A4	244	164
V	56	126	86	E5	345	229	v	76	166	118	A5	245	165
W	57	127	87	E6	346	230	w	77	167	119	A6	246	166
X	58	130	88	E7	347	231	x	78	170	120	A7	247	167
Y	59	131	89	E8	350	232	y	79	171	121	A8	250	168
Z	5A	132	90	E9	351	233	z	7A	172	122	A9	251	169
[5B	133	91	AD	255	173	{	7B	173	123	C0	300	192
\	5C	134	92	EO	340	224		7C	174	124	4F	117	79
]	5D	135	93	BD	275	189	~	7D	175	125	D0	320	208
^	5E	136	94	5F	137	95		7E	176	126	A1	241	161
_	5F	137	95	6D	155	109		7F	177	127	07	007	7

Char	ASCII			EBCDIC		
	Hex	Oct	Dec	Hex	Oct	Dec
	80	200	128	20	040	32
	81	201	129	21	041	33
	82	202	130	22	042	34
	83	203	131	23	043	35
<IND>	84	204	132	24	044	36
<NEL>	85	205	133	15	025	21
	86	206	134	06	006	6
	87	207	135	17	027	23
	88	210	136	28	050	40
	89	211	137	29	051	41
	8A	212	138	2A	052	42
	8B	213	139	2B	053	43
	8C	214	140	2C	054	44
<RI>	8D	215	141	09	011	9
	8E	216	142	0A	012	10
	8F	217	143	1B	033	27
	90	220	144	30	060	48
	91	221	145	31	061	49
	92	222	146	1A	032	26
<STS>	93	223	147	33	063	51
	94	224	148	34	064	52
	95	225	149	35	065	53
	96	226	150	36	066	54
	97	227	151	08	010	8
	98	230	152	38	070	56
	99	231	153	39	071	57
	9A	232	154	3A	072	58
<CSI>	9B	233	155	3B	073	59
<ST>	9C	234	156	04	004	4
	9D	235	157	14	024	20
	9E	236	158	3E	076	62
<APC>	9F	237	159	E1	341	225

Char	ASCII			EBCDIC		
	Hex	Oct	Dec	Hex	Oct	Dec
A0	240	160	41	101	65	
A1	241	161	42	102	66	
A2	242	162	43	103	67	
A3	243	163	44	104	68	
A4	244	164	45	105	69	
A5	245	165	46	106	70	
A6	246	166	47	107	71	
A7	247	167	48	110	72	
A8	250	168	49	111	73	
A9	251	169	51	121	81	
AA	252	170	52	122	82	
AB	253	171	53	123	83	
AC	254	172	54	124	84	
AD	255	173	55	125	85	
AE	256	174	56	126	86	
AF	257	175	57	127	87	
B0	260	176	58	130	88	
B1	261	177	59	131	89	
B2	262	178	62	142	98	
B3	263	179	63	143	99	
B4	264	180	64	144	100	
B5	265	181	65	145	101	
B6	266	182	66	146	102	
B7	267	183	67	147	103	
B8	270	184	68	150	104	
B9	271	185	69	151	105	
BA	272	186	70	160	112	
BB	273	187	71	161	113	
BC	274	188	72	162	114	
BD	275	189	73	163	115	
BE	276	190	74	164	116	
BF	277	191	75	165	117	

Char	ASCII			EBCDIC		
	Hex	Oct	Dec	Hex	Oct	Dec
C0	300	192	76	166	118	
C1	301	193	78	170	120	
C2	302	194	C2	302	194	
C3	303	195	80	200	128	
C4	304	196	8A	212	138	
C5	305	197	8B	213	139	
C6	306	198	8C	214	140	
C7	307	199	8D	215	141	
C8	310	200	8E	216	142	
C9	311	201	8F	217	143	
CA	312	202	90	220	144	
CB	313	203	9A	232	154	
CC	314	204	9B	233	155	
CD	315	205	9C	234	156	
CE	316	206	9D	235	157	
CF	317	207	9E	236	158	
D0	320	208	9F	237	159	
D1	321	209	A0	240	160	
D2	322	210	AA	252	170	
D3	323	211	AB	253	171	
D4	324	212	AC	254	172	
D5	325	213	4A	112	74	
D6	326	214	AE	256	174	
D7	327	215	AF	257	175	
D8	330	216	B0	260	176	
D9	331	217	B1	261	177	
DA	332	218	B2	262	178	
DB	333	219	B3	263	179	
DC	334	220	B4	264	180	
DD	335	221	B5	265	181	
DE	336	222	B6	266	182	
DF	337	223	B7	267	183	

Char	ASCII			EBCDIC		
	Hex	Oct	Dec	Hex	Oct	Dec
E0	340	224	B8	270	184	
E1	341	225	B9	271	185	
E2	342	226	BA	272	186	
E3	343	227	BB	273	187	
E4	344	228	BC	274	188	
E5	345	229	6A	152	106	
E6	346	230	BE	276	190	
E7	347	231	BF	277	191	
E8	350	232	CA	312	202	
E9	351	233	CB	313	203	
EA	352	234	CC	314	204	
EB	353	235	CD	315	205	
EC	354	236	CE	316	206	
ED	355	237	CF	317	207	
EE	356	238	DA	332	218	
EF	357	239	DB	333	219	
F0	360	240	DC	334	220	
F1	361	241	DD	335	221	
F2	362	242	DE	336	222	
F3	363	243	DF	337	223	
F4	364	244	EA	352	234	
F5	365	245	EB	353	235	
F6	366	246	EC	354	236	
F7	367	247	ED	355	237	
F8	370	248	EE	356	238	
F9	371	249	EF	357	239	
FA	372	250	FA	372	250	
FB	373	251	FB	373	251	
FC	374	252	FC	374	252	
FD	375	253	FD	375	253	
FE	376	254	FE	376	254	
FF	377	255	FF	377	255	